**WHAT IS CLAIMED IS:**

1. A method for implementing virtual bases with fixed offsets in a class hierarchy graph corresponding to an object oriented program, the graph having nodes representing object classes and edges representing immediate inheritance therebetween, the method comprising the steps of:

for only one node y in a set of nodes Y that directly and virtually inherit from a node x and that are not duplicated in the graph, removing an edge e that represents that the node y virtually inherits from the node x; and

adding an edge e' that represents that the node x has a fixed offset with respect to the node y.

2. The method according to claim 1, further comprising at least one of the steps of:

eliminating transitive edges in the graph; and

devirtualizing single inheritance edges in the graph.

3. The method according to claim 2, wherein said eliminating and devirtualizing steps are performed prior to said step of removing an edge e.

4. The method according to claim 3, wherein said eliminating step is performed prior to said devirtualizing step.

5. A method for implementing virtual bases with fixed offsets in a class hierarchy graph corresponding to an object oriented program, the graph having nodes representing object classes and edges representing immediate inheritance therebetween, the method comprising the steps of:

determining whether a set N is empty, the set N comprising all nodes in the graph;

removing a node x from the set N, when the set N is not empty;

determining whether a set Y is empty, the set Y comprising nodes that directly and virtually inherit from the node x;

returning to said step of determining whether the set N is empty, when the set Y is empty.

removing a node y from the set Y, when the set Y is not empty;

determining whether the node y is duplicated in the graph;

returning to said step of determining whether the set Y is empty, when the node y is duplicated; and

replacing an edge e with an edge e', when the node y is not duplicated, the edge e representing that the node y directly and virtually inherits from the node x and the edge e' representing that the node x has a fixed offset with respect to the node y.

6. The method according to claim 5, further comprising the step of returning to said step of determining whether the set N is empty, upon replacing the edge e.

7. The method according to claim 5, further comprising the step of terminating said method, when the set N is empty.

8. The method according to claim 5, wherein said step of determining whether the node y is duplicated in the graph comprises the steps of:

determining whether a set V is empty, the set V comprising nodes that directly and nonvirtually inherit from the node y;

removing a node u from the set V, when the set V is not empty;

determining whether the node u is duplicated;

determining whether a set V' is empty, when the node u is not duplicated in the graph, the set V' comprising nodes that directly inherit from the node y and that are not the node u;

returning to said step of determining whether the set V is empty, when the set V' is empty

removing a node v from the set V', when the set V' is not empty;

determining whether the node u and the node v have a common descendant;

returning to said step of determining whether the set V' is empty, when the node u and the node v do not have a common descendant;

identifying that the node y is duplicated, when one of the node u is duplicated, and the node u and the node v have a common descendant; and

identifying that the node y is not duplicated, when the set V is empty.

9. The method according to claim 8, wherein said step of determining whether the node u and the node v have a common descendant comprises the steps of:

determining whether a set L is empty, the set L comprising all the nodes in the graph;

removing a node w from the set L, when the set L is not empty;

determining whether one of the node w is the node u and the node w inherits from the node u, and whether one of the node w is the node v and the node w inherits from the node v;

returning to said step of determining whether the set L is empty, when one of the node w is not the node u and the node w does not inherit from the node u, and one of the node w is not the node v and the node w does not inherit from the node v;

identifying the nodes u and v as having a common descendant, when one of the node w is the node u and the node w inherits from the node u, and one of the node w is the node v and the node w inherits from the node v; and

identifying the nodes u and v as not having a common descendant, when the set L is empty.

10. The method according to claim 5, further comprising at least one of the steps of:

eliminating transitive edges in the graph; and

devirtualizing single inheritance edges in the graph.

11. The method according to claim 10, wherein said eliminating and devirtualizing steps are performed prior to said step of determining whether the set N is empty.

12. The method according to claim 11, wherein said eliminating step is performed prior to said devirtualizing step.

13. The method according to claim 10, wherein said eliminating step comprises the steps of:

determining whether a set M is empty, the set M comprising the nodes in the graph;

removing a node m from the set M, when the set M is not empty;

determining whether a set M' is empty, the set M' comprising nodes that directly and virtually inherit from the node m;

returning to said step of determining whether the set M is empty, when the set M' is empty;

removing a node m' from the set M', when the set M' is not empty;

determining whether a set M" is empty, the set M" comprising the nodes in the set M' other than the node m';

returning to said step of determining whether the set M' is empty, when the set M" is empty;

removing a node z from the set M", when the set M" is not empty;

determining whether the node m' is a base of the node z and not the node z;

removing an edge e1 from the graph, when the node m' is a base of the node z and not the node z, the edge e1 representing that the node z virtually inherits from the node m;

determining whether the node z is a base of the node m' and not the node m', when the node m' is not a base of the node z and is the node z;

removing an edge e2 from the graph, when the node z is a base of the node m' and not the node m', the edge e2 representing that the node m' virtually inherits from the node m; and

returning to said step of determining whether the set M" is empty, upon one of removing the edge e1, removing the edge e2, and when the node z is not the node m' and a base of the node m'.

14. The method according to claim 10, wherein said devirtualizing step comprises the steps of:

determining whether a set P is empty, the set P comprising all the nodes in the graph;

removing a node p from the set P, when the set P is not empty;

determining whether a set S is empty, the set S comprising nodes that directly and virtually inherit from the node p and that are not duplicated in the graph;

returning to said step of determining whether the set P is empty, when the set S is empty;

removing a node s from the set S, when the set S is not empty;

determining whether a set S' is empty, the set S' comprising the nodes in the set S except the node s;

removing a node s' from the set S', when the set S' is not empty;

determining whether the node s and the node s' have a common descendant;

returning to said step of determining whether the set S' is empty, when the node s and the node s' do not have the common descendant;

replacing an edge e with an edge e', when the set S' is empty, the edge e representing that the node s directly and virtually inherits from the node p, and the edge e'

representing that the node s has a fixed offset with respect to the node p; and

returning to said step of determining whether the set S is empty, upon one of determining that the node s and the node s' have the common descendant and replacing the edge e.

15. A method for implementing virtual bases with fixed offsets in a class hierarchy graph corresponding to an object oriented program, the graph having nodes representing object classes and edges representing immediate inheritance therebetween, the method comprising the steps of:

determining whether a set V' is empty, the set V' comprising nodes that virtually inherit from a node v in the graph;

removing a node u from the set V', when the set V' is not empty;

determining whether the node u is duplicated;

adding the node u to a set V, when the node u is not duplicated in the graph, the set V initially being an empty set of nodes that directly and virtually inherit from the node v and that are not duplicated in the graph;

returning to said step of determining whether the set V' is empty, upon one of adding the node u to the set V and when the node u is duplicated;

determining whether the set V is empty;

selecting a subset S of the set V such that the subset S is a maximal independent set in a set G, when the set V is not empty, the set G comprising a first ordered pair of the set V and a set E, the set E comprising a second ordered pair of a node u1 and a node u2, the nodes u1 and u2 included in the set V and having a common descendant in the graph;

determining whether the subset S is empty;

removing a node s from the subset S, when the subset S is not empty; and

replacing an edge e with an edge e', the edge e representing that the node s virtually inherits from the node v, the edge e' representing that the node v has a fixed offset with respect to the node s.

16. The method according to claim 15, further comprising the step of returning to said step of determining whether the subset S is empty, upon replacing the edge e.

17. The method according to claim 15, further comprising the step of terminating said method, when one of the set V and the subset S is empty.

18. The method according to claim 15, further comprising the step of eliminating transitive edges in the graph.

19. The method according to claim 18, wherein said eliminating step is performed prior to said step of determining whether the set N is empty.

20. The method according to claim 18, wherein said eliminating step comprises the steps of:

determining whether a set Q is empty, the set Q comprising the nodes in the graph;

removing a node x from the set Q, when the set Q is not empty;

determining whether a set Q' is empty, the set Q' comprising nodes that directly and virtually inherit from the node x;

returning to said step of determining whether the set Q is empty, when the set Q' is empty;

removing a node y from the set Q', when the set Q' is not empty;

determining whether a set Q" is empty, the set Q" comprising the nodes in the set Q' other than the node y;

returning to said step of determining whether the set Q' is empty, when the set Q" is empty;

removing a node z from the set Q", when the set Q" is not empty;

determining whether the node y is a base of the node z and not the node z;

removing an edge e1 from the graph, when the node y is a base of the node z and not the node z, the edge e1 representing that the node z virtually inherits from the node x;

determining whether the node z is a base of the node y and not the node y, when the node y is not a base of the node z and is the node z;

removing an edge e2 from the graph, when the node z is a base of the node y and not the node y, the edge e2 representing that the node y virtually inherits from the node x; and

returning to said step of determining whether the set Q" is empty, upon one of removing the edge e1, removing the edge e2, and when the node z is a base of the node y and not the node y.